array – Manipulate array variables

SYNOPSIS

array option arrayName ?arg arg ...?

DESCRIPTION

This command performs one of several operations on the variable given by arrayName. ArrayName must be the name of an existing array variable. The option argument determines what action is carried out by the command. The legal options (which may be abbreviated) are:

array anymore arrayName searchId
: Returns 1 if there are any more elements left to be processed in an array search, 0 if all elements have already been returned. SearchId indicates which search on arrayName to check, and must have been the return value from a previous invocation of array startsearch. This option is particularly useful if an array has an element with an empty name, since the return value from array nextelement won't indicate whether the search has been completed.

array donesearch arrayName searchId
: This command terminates an array search and destroys all the state associated with that search. SearchId indicates which search on arrayName to destroy, and must have been the return value from a previous invocation of array startsearch. Returns an empty string.

array names arrayName
: Returns a list containing the names of all of the elements in the array. If there are no elements in the array then an empty string is returned.

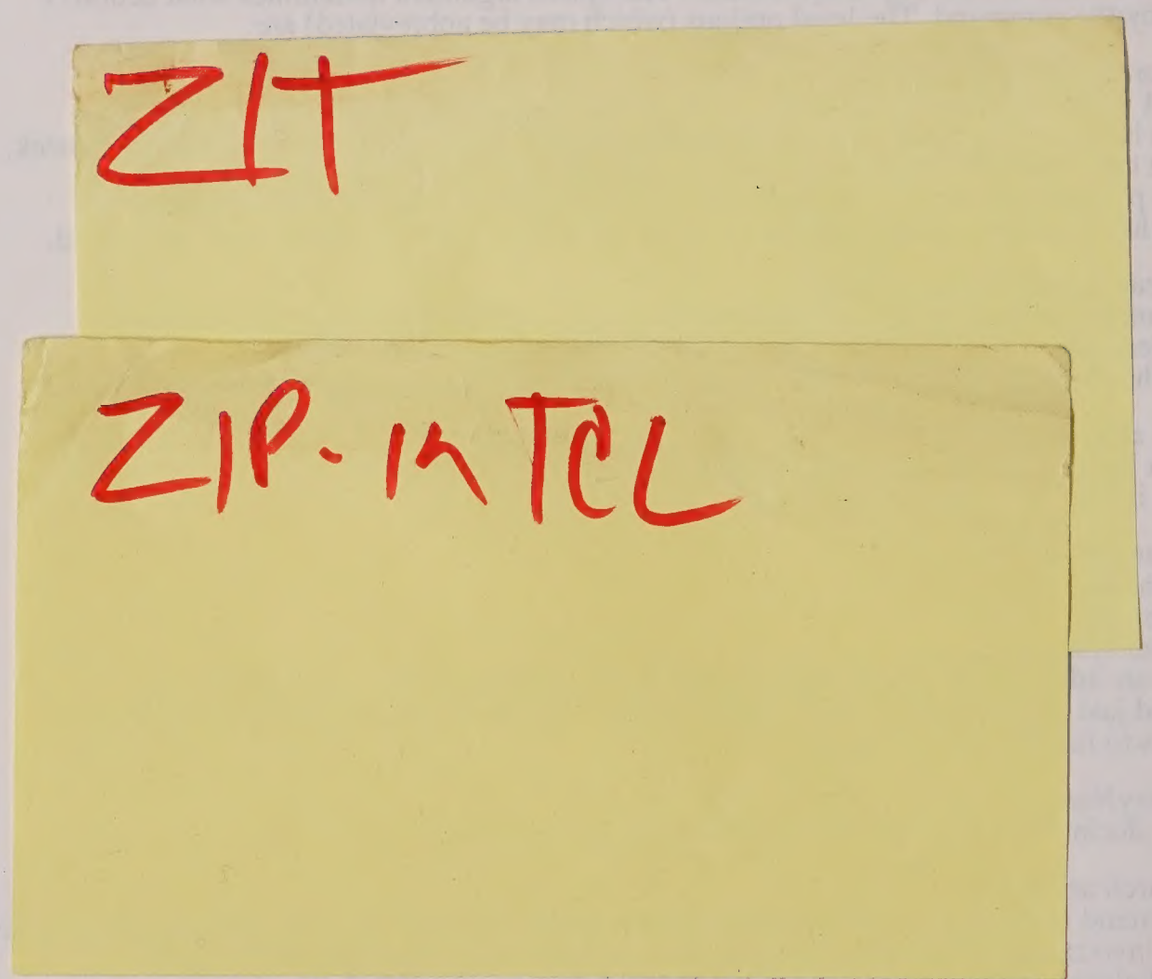
array nextelement arrayName searchId
: Returns the name of the next element in arrayName, or an empty string if all elements of arrayName have already been returned in this search. The searchId argument identifies the search, and must have been the return value of an array startsearch command. Warning: if elements are added to or deleted from the array, then all searches are automatically terminated just as if array donesearch had been invoked; this will cause array nextelement operations to fail for those searches.

array size arrayName
: Returns a decimal string giving the number of elements in the array.

array startsearch arrayName
: This command initializes an element–by–element search through the array given by arrayName, such that invocations of the array nextelement command will return the names of the individual elements in the array. When the search has been completed, the array donesearch command should be invoked. The return value is a search identifier that must be used in array nextelement and array donesearch commands; it allows multiple searches to be underway simultaneously for the same array.

KEYWORDS

array, element names, search


ZIT
ZIP-IN TCL

tclsh – Simple shell containing Tcl interpreter

SYNOPSIS

tclsh ?fileName arg arg ...?

DESCRIPTION

Tclsh is a shell–like application that reads Tcl commands from its standard input or from a file and evaluates them. If invoked with no arguments then it runs interactively, reading Tcl commands from standard input and printing command results and error messages to standard output. It runs until the exit command is invoked or until it reaches end–of–file on its standard input. If there exists a file .tclshrc in the home directory of the user, tclsh evaluates the file as a Tcl script just before reading the first command from standard input.

SCRIPT FILES

If tclsh is invoked with arguments then the first argument is the name of a script file and any additional arguments are made available to the script as variables (see below). Instead of reading commands from standard input tclsh will read Tcl commands from the named file; tclsh will exit when it reaches the end of the file. There is no automatic evaluation of .tclshrc in this case, but the script file can always source it if desired.

If you create a Tcl script in a file whose first line is

```
#!/usr/local/bin/tclsh
```

then you can invoke the script file directly from your shell if you mark the file as executable. This assumes that **tclsh** has been installed in the default location in /usr/local/bin; if it's installed somewhere else then you'll have to modify the above line to match.

VARIABLES

Tclsh sets the following Tcl variables:

argc
: Contains a count of the number of arg arguments (0 if none), not including the name of the script file.

argv
: Contains a Tcl list whose elements are the arg arguments, in order, or an empty string if there are no arg arguments.

argv0
: Contains fileName if it was specified. Otherwise, contains the name by which tclsh was invoked.

tcl_interactive
: Contains 1 if tclsh is running interactively (no fileName was specified and standard input is a terminal–like device), 0 otherwise.

PROMPTS

string – Manipulate strings

SYNOPSIS

string option arg ?arg ...?

DESCRIPTION

Performs one of several string operations, depending on option. The legal options (which may be abbreviated) are:

string compare string1 string2
Perform a character–by–character comparison of strings string1 and string2 in the same way as the C strcmp procedure. Return –1, 0, or 1, depending on whether string1 is lexicographically less than, equal to, or greater than string2.

string first string1 string2
Search string2 for a sequence of characters that exactly match the characters in string1. If found, return the index of the first character in the first such match within string2. If not found, return –1.

string index string charIndex
Returns the charIndex'th character of the string argument. A charIndex of 0 corresponds to the first character of the string. If charIndex is less than 0 or greater than or equal to the length of the string then an empty string is returned.

string last string1 string2
Search string2 for a sequence of characters that exactly match the characters in string1. If found, return the index of the first character in the last such match within string2. If there is no match, then return –1.

string length string
Returns a decimal string giving the number of characters in string.

string match pattern string
See if pattern matches string; return 1 if it does, 0 if it doesn't. Matching is done in a fashion similar to that used by the C–shell. For the two strings to match, their contents must be identical except that the following special sequences may appear in pattern:

Matches any sequence of characters in string, including a null string.
Matches any single character in string.
Matches any character in the set given by chars. If a sequence of the form x–y appears in chars, then any character between x and y, inclusive, will match.
Matches the single character x. This provides a way of avoiding the special interpretation of the characters *?[]\ in pattern.

string range string first last
Returns a range of consecutive characters from string, starting with the character whose index is first and ending with the character whose index is last. An index of 0 refers to the first character of the string. Last may be end (or any abbreviation of it) to refer to the last character of the string. If first is less than zero then it is treated as if it were zero, and if last is greater than or equal to the length of the string then it is treated as if it were end. If first is greater than last then an empty string is returned.

string tolower string
Returns a value equal to string except that all upper case letters have been converted to lower case.

list – Create a list

SYNOPSIS

list ?arg arg ...?

DESCRIPTION

This command returns a list comprised of all the args, or an empty string if no args are specified. Braces and backslashes get added as necessary, so that the index command may be used on the result to re–extract the original arguments, and also so that eval may be used to execute the resulting list, with arg1 comprising the command's name and the other args comprising its arguments. List produces slightly different results than concat: concat removes one level of grouping before forming the list, while list works directly from the original arguments. For example, the command

```
list a b {c d e} {f {g h}}
```

will return

```
a b {c d e} {f {g h}}
```

while **concat** with the same arguments will return

```
a b c d e f {g h}
```

KEYWORDS

element, list

array – Manipulate array variables

SYNOPSIS

array option arrayName ?arg arg ...?

DESCRIPTION

This command performs one of several operations on the variable given by arrayName. ArrayName must be the name of an existing array variable. The option argument determines what action is carried out by the command. The legal options (which may be abbreviated) are:

array anymore arrayName searchId
: Returns 1 if there are any more elements left to be processed in an array search, 0 if all elements have already been returned. SearchId indicates which search on arrayName to check, and must have been the return value from a previous invocation of array startsearch. This option is particularly useful if an array has an element with an empty name, since the return value from array nextelement won't indicate whether the search has been completed.

array donesearch arrayName searchId
: This command terminates an array search and destroys all the state associated with that search. SearchId indicates which search on arrayName to destroy, and must have been the return value from a previous invocation of array startsearch. Returns an empty string.

array names arrayName
: Returns a list containing the names of all of the elements in the array. If there are no elements in the array then an empty string is returned.

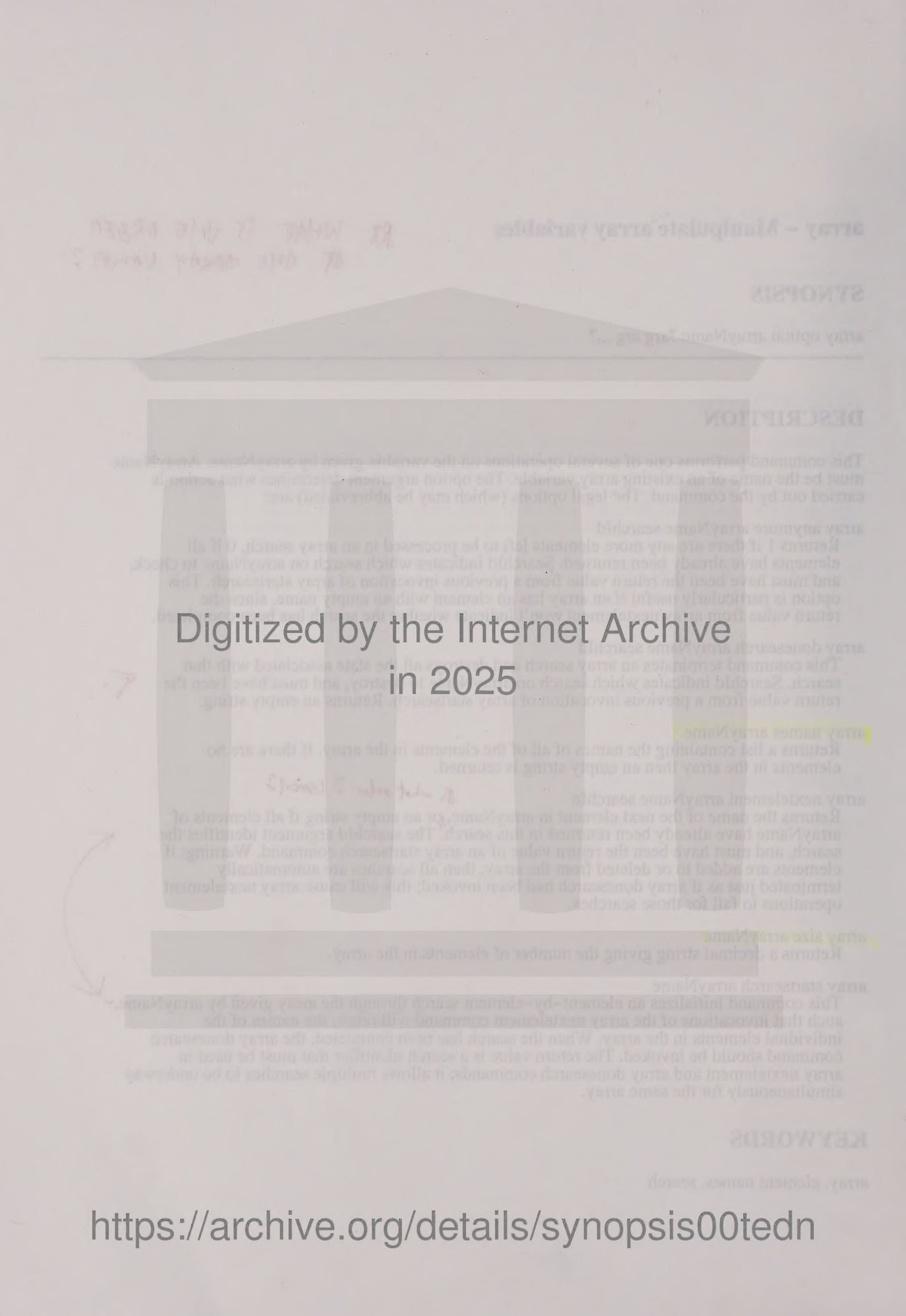
array nextelement arrayName searchId
: Returns the name of the next element in arrayName, or an empty string if all elements of arrayName have already been returned in this search. The searchId argument identifies the search, and must have been the return value of an array startsearch command. Warning: if elements are added to or deleted from the array, then all searches are automatically terminated just as if array donesearch had been invoked; this will cause array nextelement operations to fail for those searches.

array size arrayName
: Returns a decimal string giving the number of elements in the array.

array startsearch arrayName
: This command initializes an element–by–element search through the array given by arrayName, such that invocations of the array nextelement command will return the names of the individual elements in the array. When the search has been completed, the array donesearch command should be invoked. The return value is a search identifier that must be used in array nextelement and array donesearch commands; it allows multiple searches to be underway simultaneously for the same array.

KEYWORDS

array, element names, search


Digitized by the Internet Archive
in 2025

https://archive.org/details/synopsis00tedn

linsert – Insert elements into a list

SYNOPSIS

linsert list index element ?element element ...?

DESCRIPTION

This command produces a new list from list by inserting all of the element arguments just before the indexth element of list. Each element argument will become a separate element of the new list. If index is less than or equal to zero, then the new elements are inserted at the beginning of the list. If index is greater than or equal to the number of elements in the list, then the new elements are appended to the list.

KEYWORDS

element, insert, list

linsert - Insert elements into a list

SYNOPSIS

linsert list index element ?element ...?

DESCRIPTION

This command produces a new list from list by inserting all of the element arguments just before the index element of list. Each element argument will become a separate element of the new list. If index is less than or equal to zero, then the new elements are inserted at the beginning of the list. If index is greater than or equal to the number of elements in the list, then the new elements are appended to the list.

KEYWORDS

element, insert, list

string toupper string
Returns a value equal to string except that all lower case letters have been converted to upper case.

string trim string ?chars?
Returns a value equal to string except that any leading or trailing characters from the set given by chars are removed. If chars is not specified then white space is removed (spaces, tabs, newlines, and carriage returns).

string trimleft string ?chars?
Returns a value equal to string except that any leading characters from the set given by chars are removed. If chars is not specified then white space is removed (spaces, tabs, newlines, and carriage returns).

string trimright string ?chars?
Returns a value equal to string except that any trailing characters from the set given by chars are removed. If chars is not specified then white space is removed (spaces, tabs, newlines, and carriage returns).

KEYWORDS

case conversion, compare, index, match, pattern, string

split – Split a string into a proper Tcl list

SYNOPSIS

split string ?splitChars?

DESCRIPTION

Returns a list created by splitting string at each character that is in the splitChars argument. Each element of the result list will consist of the characters from string that lie between instances of the characters in splitChars. Empty list elements will be generated if string contains adjacent characters in splitChars, or if the first or last character of string is in splitChars. If splitChars is an empty string then each character of string becomes a separate element of the result list. SplitChars defaults to the standard white–space characters. For example,

```
split "comp.unix.misc" .
```

returns "comp unix misc" and

```
split "Hello world" {}
```

returns "H e l l o { } w o r l d".

KEYWORDS

list, split, string

When tclsh is invoked interactively it normally prompts for each command with "% ". You can change the prompt by setting the variables tcl_prompt1 and tcl_prompt2. If variable tcl_prompt1 exists then it must consist of a Tcl script to output a prompt; instead of outputting a prompt tclsh will evaluate the script in tcl_prompt1. The variable tcl_prompt2 is used in a similar way when a newline is typed but the current command isn't yet complete; if tcl_prompt2 isn't set then no prompt is output for incomplete commands.

KEYWORDS

argument, interpreter, prompt, script file, shell

lindex – Retrieve an element from a list

SYNOPSIS

lindex list index

DESCRIPTION

This command treats list as a Tcl list and returns the index'th element from it (0 refers to the first element of the list). In extracting the element, lindex observes the same rules concerning braces and quotes and backslashes as the Tcl command interpreter; however, variable substitution and command substitution do not occur. If index is negative or greater than or equal to the number of elements in value, then an empty string is returned.

KEYWORDS

element, index, list

lindex - Retrieve an element from a list

SYNOPSIS

lindex list index

DESCRIPTION

This command treats list as a Tcl list and returns the index'th element from it (0 refers to the first element of the list). In extracting the element, lindex observes the same rules concerning braces and quotes and backslashes as the Tcl command interpreter; however, variable substitution and command substitution do not occur. If index is negative or greater than or equal to the number of elements in value, then an empty string is returned.

KEYWORDS

element, index, list

lsearch – See if a list contains a particular element

SYNOPSIS

lsearch ?mode? list pattern

DESCRIPTION

This command searches the elements of list to see if one of them matches pattern. If so, the command returns the index of the first matching element. If not, the command returns –1. The mode argument indicates how the elements of the list are to be matched against pattern and it must have one of the following values:

–exact
: The list element must contain exactly the same string as pattern.

–glob
: Pattern is a glob–style pattern which is matched against each list element using the same rules as the string match command.

–regexp
: Pattern is treated as a regular expression and matched against each list element using the same rules as the regexp command.

If mode is omitted then it defaults to –glob.

KEYWORDS

list, match, pattern, regular expression, search, string